Operations Research
Vol. 53, No. 1, January–February 2005, pp. 48–60
issn 0030-364X | eissn 1526-5463 | 05 | 5301 | 0048

doi 10.1287/opre.1040.0140
© 2005 INFORMS

Stochastic Transportation-Inventory Network Design Problem

Jia Shu, Chung-Piaw Teo
High Performance Computation for Engineered Systems, Singapore-MIT Alliance, and Department of Decision Sciences,
National University of Singapore, Singapore {tlisj@nus.edu.sg, bizteocp@nus.edu.sg}

Zuo-Jun Max Shen
Department of Industrial Engineering and Operations Research, University of California, Berkeley, California 94720,
shen@ieor.berkeley.edu

We study the stochastic transportation-inventory network design problem involving one supplier and multiple retailers. Each retailer faces some uncertain demand, and safety stock must be maintained to achieve suitable service levels. However, risk-pooling benefits may be achieved by allowing some retailers to serve as distribution centers for other retailers. The problem is to determine which retailers should serve as distribution centers and how to allocate the other retailers to the distribution centers. Shen et al. (2003) formulated this problem as a set-covering integer-programming model. The pricing problem that arises from the column generation algorithm gives rise to a new class of the submodular function minimization problem. In this paper, we show that by exploiting certain special structures, we can solve the general pricing problem in Shen et al. efficiently. Our approach utilizes the fact that the set of all lines in a two-dimension plane has low VC-dimension. We present computational results on several instances of sizes ranging from 40 to 500 retailers. Our solution technique can be applied to a wide range of other concave cost-minimization problems.

Subject classifications: facilities/equipment planning: stochastic; inventory/production: uncertainty, stochastic; programming: nonlinear.
Area of review: Transportation.
History: Received October 2001; revision received December 2002; accepted September 2003.

1. Introduction

Managing inventory has become a major challenge for companies as they simultaneously try to reduce costs and improve service levels in today's increasingly competitive market. Managing inventory consists of two critical tasks. First, we must determine the optimum number and location of distribution centers. Second, we must determine the amount of inventory to maintain at each of the distribution centers. Often these tasks are undertaken separately, resulting in a degree of suboptimization. Research in this area deals with the modelling, design, planning, and control of integrated supply chains as they occur in both industrial and service organizations. Important aspects concern the development of new control architectures, as well as the development of decision-support systems for planning and scheduling these systems. In addition, methods for the performance analysis of alternative supply chain structures, under stochastic demand, is an important research area. The current research seeks to optimally exploit the possibilities to achieve an ideal network that must have the optimum number, size, and location of DCs to support the inventory replenishment activities of its retailers.

We study the design of a stochastic distribution network in which a single supplier ships products to a set of distribution centers (DCs). Each DC serves a pool of retailers with uncertain customer demand. The number and locations of DCs are not given a priori. They are chosen from the set of retailers. After being chosen as a DC, this retailer-based DC is served directly by the supplier and distributes products to other retailers assigned to it. The central issue in the stochastic distribution network design problem is how many and which retailers should be selected to be the DCs, how to assign the other retailers to the DCs, and how to manage the inventory at each DC.

The problem we presented above was motivated by a study conducted by Shen et al. (2003) and Daskin et al. (2002) at a Chicago-based blood bank. The blood bank supplied roughly 30 hospitals in the greater Chicago area. Its focus was on the production and distribution of platelets, the most expensive and most perishable of all blood products. If a unit of platelets is not used within five days of the time it is produced from whole blood, it must be destroyed. The demand for platelets is highly variable, as they are needed in only a limited number of medical contexts. When they are used, however, multiple units are often needed. By storing platelets at regional centers (located at a subset of the hospitals) instead, and distributing platelets to nearby hospitals on an as-needed or daily basis, three objectives were likely to be achievable:

- Inventory cost can be reduced due to risk pooling. The safety stocks needed to protect against shortages will reduce with pooling of safety stocks;

- The cost of emergency shipments can be reduced because platelets would be stored closer to each of the hospitals; and
- The training cost for inventory managers in an improved supply chain distribution network system can be reduced because the stock would be stored at a small number of regional distribution centers instead of being maintained at each individual hospital.

Shen et al. (2003) and Daskin et al. (2002) modelled the above problem as a special case of a more general concave cost network design problem, often encountered in practice. For instance, consider the following case problem from a widely used supply chain management textbook by Chopra and Meindl (2001). ALKO Inc., a company that developes, produces, and distributes lighting fixtures, made over 100 products through its production line. These products were stored in five regional DCs operated by ALKO to meet the market demand nationwide. It classified the products into three categories in terms of volume of sales, and each one has a different demand mean and standard deviation. The company wanted to determine whether it should consolidate all or some of its products into a central DC and close all or some of the regional DCs, instead of stocking each item in every regional DC. The decision as to whether to set up the central DC depends on:

- setup cost of a central DC if any;
- closure cost of regional DCs if any; and
- cycle inventory cost, safety stock cost, and transportation cost under the new distribution system.

The setup cost of a central DC is modelled as a concave cost function of the throughput of the DC. It depends on the products stored at the DC and the regions served. The associated inventory cost and safety inventory cost components can be well approximated by another concave function, depending on the DC-customer assignment decision and the inventory replenishment policy used. This model, interestingly, turns out to be an (multiproduct) extension of the concave cost network design problem studied in this paper.

The rest of this paper is organized as follows. Section 2 reviews some literature on location theory and inventory models, and some earlier work on joint location-inventory models. Section 3 describes two models for the network design problem, namely the nonlinear integer programming model and the set covering model. In §4, we present a new method to solve the nonlinear pricing problem. In §5, we use the "variable fixing" method to further enhance the computational performance of the column generation algorithm. Finally, computational results that highlight the effectiveness of the proposed algorithm are reported in §6. We conclude the paper with several ways to extend and generalize the approach presented in §7.

2. Literature Review

The goal of traditional inventory research (see, for example, Graves et al. 1993, Nahmias 1997, and Zipkin 1997 for a review) is to develop and evaluate inventory policies so as to minimize the inventory-related costs while meeting some service-level standards. Most of the papers in this area assumed a given distribution structure, with given DC location and known customers assignment to the DC. In a different vein, the literature on facility location and distribution network design (see, for example, Daskin 1995, Mirchandani and Francis 1990, Drezner 1995, and Geoffrion and Power 1995 for a review) focuses mainly on the trade-offs between the facility location and product transportation costs, usually ignoring or simplifying inventory-related costs.

Eppen (1979) showed that significant inventory-cost savings can be achieved at the network design stage by grouping retailers demand together and thus capitalizing on the so-called "risk-pooling effects." The location issue is therefore an important factor in the overall performance of distribution inventory system. Barahona and Jensen (1998) studied a version of the distribution network design problem for computer spare parts. Their model takes into account the costs of building the DCs and maintaining inventories at the various locations. To make the model tractable, they imposed very restrictive assumptions on the inventory costs. Teo et al. (2001) studied the impact on inventory costs with consolidation of distribution centers. They design an algorithm that solves for a distribution system with the total fixed facility location cost and inventory costs within $\sqrt{2}$ of the optimum. Their approach, however, could not capture the impact of network design on the transportation cost component. Erlebacher and Meller (2000) formulated a joint location-inventory model with highly nonlinear integer objective functions. Continuous approximation and some other heuristics are used to solve the problem. For a 600-node problem, it took 117 hours on a Sun Ultra Sparcstation.

Finally, Shen (2000), Shen et al. (2003), and Daskin et al. (2002) studied the risk-pooling network design problem (presented in the next section). They were able to solve the pricing problem efficiently in time $O(n \log n)$ for two special cases: when the variance of the demand is proportional to the mean (as in the Poisson demand case), or when the demand is deterministic. Unfortunately, they are unable to address the general pricing problem. Although they proved that the general pricing problem is a submodular function minimization problem that can be readily solved in polynomial time (see, for example, Grotschel et al. 1981, Schrijver 2000, Iwata et al. 2001), preliminary computational evidence shows that these algorithms are still not attractive computationally, especially because we need to solve the pricing problem in every iteration of the column generation algorithm.

Fortunately, the specific properties of the general pricing problem make possible the use of a considerably simpler and faster algorithm. We propose a much faster algorithm ($O(n^2 \log n)$) to generate columns for the general case, using ideas from Chakravarty et al. (1985). We show that the pricing problem is related to a problem in computational geometry. In fact, using an advance incremental algorithm for enumeration of vertices over a zonotope (see Onn and Schulman 2001, where the pricing problem in this paper can be shown to be a special case), the running time complexity can be further slashed to $O(n^2)$. However, the reduction in running time comes at the expense of more complicated data structure to implement the incremental algorithm. The proposed algorithm, thus, is still not attractive in practice. Instead, we show that by combining the variable fixing idea (cf., Daskin et al. 2002) with our algorithm for the pricing problem, we are able to speed up the running time by a factor of 10 and solve a realistic-sized transportation-inventory network design problem (up to 500 retailers) in just under 10 minutes.

We emphasize the importance of being able to address the general pricing problem efficiently. The two cases considered in Shen et al. (2003) require that the demand be either deterministic or $\sigma_i^2/\mu_i = \gamma$ for every retailer. This is hardly the case in many real situations. Supply chain network design problems under the more general conditions are the ones that management is most concerned with, and our model can be applied in the decision-making process for problems of this kind.

While the development of an integrated location/inventory model will be valuable in its own right, it is likely to have significant impacts in other areas as well. In a recent paper, Current et al. (2002) cite over a dozen applications of location models in many different areas including chip manufacturing, medical diagnosis, product procurement, and lot-sizing problems. Some of these problems are structurally similar to our integrated location/inventory model. Thus, there are likely to be significant applications of this work that extend well beyond integrated facility location and inventory modelling and beyond supply chain network design. One such example is Geunes et al. (2004), who apply the algorithm in Shen et al. (2003) to an inventory model in which a seller maximizes profit by deciding whether or not to satisfy each potential market demand. The decision depends on the revenue and cost parameters of each individual market and the economies of scale in the production/distribution costs.

3. Model Formulation

The generic risk-pooling network design problem is as follows: Given a set I of retailers, we would like to (i) determine the location of the DCs, and (ii) determine how the DC can be used to serve the retailers. We use the following notations:

Inputs and Parameters

- μ_i: mean (yearly) demand at retailer i for each $i \in I$.
- σ_i^2: variance of (daily) demand at retailer i for each $i \in I$.
- f_j: fixed (annual) cost of locating a regional distribution center at retailer j for each $j \in I$.

We further assume that each retailer can only be served using a single DC; i.e., we do not split the demand at the retailer. When DC j is used to serve the retailers in set S, the associated cost is given by four cost components:

Cost Components

- f_j: the fixed location cost.
- $\sum_{i\in S} \alpha_{ij}\mu_i$: a term which is *linear* in μ_i.
- $G_j(\sum_{i\in S} \mu_i)$: a term which is *concave* and *nondecreasing* in the *expected throughput* assigned.
- $H_j(\sum_{i\in S} \sigma_i^2)$: a term which is *concave* and *nondecreasing* in the *total demand variance* experienced by the DC.

For instance, when α_{ij} corresponds to unit transportation cost between DC j and retailer i, the second term captures the total transportation cost associated with using DC j to serve retailers in set S. $G_j(\sum_{i\in S} \mu_i)$ can be interpreted as the DC operating and inventory replenishment cost. This is normally assumed to be concave (as in the ALKO case or as in the EOQ model approximation) to reflect the economy of scales in inventory replenishment and handling. Last but not least, $H_j(\sum_{i\in S} \sigma_i^2)$ can be interpreted as the safety inventory cost component associated with the assignment. Again, this function can be well approximated as a concave function in the total variance in the demand assigned to the DC, capturing the risk-pooling effect by consolidating demand at a centralized location.

Let $X_j = 1$ if a DC is set up at location j, 0 otherwise, and $Y_{i,j} = 1$ if DC j is used to serve retailer i, 0 otherwise.

The problem can be formulated as

$$\min \sum_{j\in I}\Bigg(f_j X_j + \bigg(\sum_{i\in I} (\alpha_{ij}\mu_i) Y_{ij}\bigg) + G_j\bigg(\sum_{i\in I} \mu_i Y_{ij}\bigg) + H_j\bigg(\sum_{i\in I} \sigma_i^2 Y_{ij}\bigg)\Bigg) \tag{1}$$

subject to

$$\sum_{j\in I} Y_{ij} = 1 \quad \text{for each } i \in I, \tag{2}$$

$$Y_{ij} - X_j \leqslant 0 \quad \text{for each } i, j \in I, \tag{3}$$

$$Y_{ij} \in \{0, 1\} \quad \text{for each } i, j \in I, \tag{4}$$

$$X_j \in \{0, 1\} \quad \text{for each } j \in I. \tag{5}$$

The first two terms are structurally identical to those of the uncapacitated facility model. The last two terms are related to inventory costs, which are nonlinear in the assignment variables. The constraints of the model are identical to those of the uncapacitated facility location problem, thus the problem we are studying is more difficult than the standard uncapacitated facility location problem, which is already a notorious NP-hard problem. Note that without the risk-pooling term $H_j(\cdot)$, the LP relaxation of the above problem reduces to a classical concave cost transportation network flow problem. The presence of the risk-pooling term destroys the network flow structure and gives rise to a nasty combinatorial optimization problem.

3.1. Example of the Risk-Pooling Model

For ease of exposition and to make the paper self-contained, we introduce the model proposed by Shen et al. (2003) and Daskin et al. (2002) as a special case of the above model. The readers may want to refer to the original papers for detailed derivation of the model.

Two different types of inventories are kept at each DC: the working inventory, which is determined by the inventory ordering policy adopted, and the safety stock, which is kept at each DC to protect against the possibilities of running out of stocks during replenishment lead time. We assume each DC orders inventory from the supplier using an economic order quantity model (EOQ). Other cost terms include the transportation costs from each DC to the retailers it serves and the level of safety stock to maintain, which are also dependent on the decisions of retailer assignments.

Following another assumption made in Shen et al. (2003), we assume that the non-DC retailers maintain only a minimal amount of inventory, and we therefore ignore this inventory in the model below.

To model this problem, we define the following additional notation:

Additional Inputs and Parameters

- d_{ij}: cost per unit to ship from retailer j to retailer i for each $i \in I$ and $j \in I$.
- α: desired percentage of retailers' orders satisfied (fill rate).
- β: weight factor associated with the shipment cost.
- θ: weight factor associated with the inventory cost.
- z_α: standard normal deviate such that $P(z \leqslant z_\alpha) = \alpha$.
- h: inventory holding cost per unit of product per year.
- F_j: fixed cost of placing an order at distribution center j for each $j \in I$.
- L: lead time in days.
- g_j: fixed shipment cost from external supplier to distribution center j.
- a_j: per unit shipment cost from external supplier to distribution center j.

Note that to simplify the notation, we have assumed that all lead times are equal. In this case, the model reduced to

$$\min \sum_{j\in I}\left(f_j X_j + \left[\sum_{i\in I}(\beta\mu_i d_{ij} + \beta a_j \mu_i) Y_{ij}\right] + \sqrt{2\theta h(F_j+\beta g_j)}\sqrt{\sum_{i\in I}\mu_i Y_{ij}} + \theta h z_\alpha \sqrt{L}\sqrt{\sum_{i\in I}\sigma_i^2 Y_{ij}}\right)$$

$$\equiv \sum_{j\in I}\left(f_j X_j + \left(\sum_{i\in I}\hat{d}_{ij} Y_{ij}\right) + K_j\sqrt{\sum_{i\in I}\mu_i Y_{ij}} + q\sqrt{\sum_{i\in I}\sigma_i^2 Y_{ij}}\right) \tag{6}$$

subject to

$$\sum_{j\in I} Y_{ij} = 1 \quad \text{for each } i \in I, \tag{7}$$

$$Y_{ij} - X_j \leqslant 0 \quad \text{for each } i, j \in I, \tag{8}$$

$$Y_{ij} \in \{0, 1\} \quad \text{for each } i, j \in I, \tag{9}$$

$$X_j \in \{0, 1\} \quad \text{for each } j \in I, \tag{10}$$

where

$$\hat{d}_{ij} = \beta\mu_i(d_{ij} + a_j),$$

$$K_j = \sqrt{2\theta h(F_j + \beta g_j)},$$

$$q = \theta h z_\alpha \sqrt{L}.$$

The objective function minimizes the weighted sum of the following four cost components:

- The fixed cost of locating facilities, given by the term $\sum_j f_j X_j$.
- The annual shipment cost from the distribution centers to the non-DC retailers, given by the term $\beta(\sum_{i\in I}(\mu_i d_{ij} + a_j\mu_i)Y_{ij})$.
- The expected working inventory cost, given the solution to the EOQ equation with ordering cost $F_j + \beta g_j$, holding cost θh, and demand $\sum_{i\in I}\mu_i Y_{ij}$.
- The annual safety stock cost, given by $\theta h z_\alpha \sqrt{L} \cdot \sqrt{\sum_{i\in I}\sigma_i^2 Y_{ij}}$.

This is easily seen to be a special case of the generic risk-pooling network design problem presented earlier.

3.2. The Set-Covering Formulation

The nonlinear integer-programming model proposed earlier is normally approximated by converting the problem to a linear integer-programming problem, through linearizing the objective function (cf., Erlebacher and Meller 2000). We can also solve the proposed nonlinear integer-programming problem directly using the Lagrangian relaxation approach (cf., Daskin et al. 2002). We propose in the rest of this section a different but equivalent way to study the generic risk-pooling network design problem. The advantage of this approach is that it gets rid of the nonlinearity (using exponentially many variables) inherent in the previous model, and it is known to be equivalent ("dual") to the Lagrangian relaxation approach.

Note that every feasible solution to our decision problem consists of a partition of the set I of retailers into nonempty subsets, $R_1, R_2, \ldots, R_n$, together with one designated DC for each of these n sets.

Let $\mathcal{R}$ be the collection of all nonempty subsets of the set I. Let $c_{R,j}$ be the total cost associated with DC j serving the set of retailers in R. That is,

$$c_{R,j} = f_j + \sum_{i\in R}\alpha_{ij} + G_j\left(\sum_{i\in R}\mu_i\right) + H_j\left(\sum_{i\in R}\sigma_i^2\right).$$

Let $z_{R,j} = 1$ if DC j is used to serve the set of retailers in R. The set-covering model for the network design problem can now be formulated as

$$(\mathcal{M}_{\mathcal{R}}) \quad \min \sum_{R\in\mathcal{R}}\sum_{j\in I} c_{R,j} z_{R,j}$$

subject to

$$\sum_{R\in\mathcal{R}:\, i\in R}\left(\sum_{j\in I} z_{R,j}\right)\geqslant 1 \quad \forall i\in I,$$

$$z_{R,j}\in\{0,1\} \quad \forall R\in\mathcal{R}.$$

We begin each iteration by solving the linear relaxation of the above set-covering model, using a partial set of columns, to obtain an optimal solution $\bar{z}_{R,j}$, $R\in\mathcal{R}$, and the corresponding optimal dual solution $\bar{\pi}_i$, $i\in I$.

We want to know for each column (R,j), whether the reduced cost

$$c_{R,j}-\sum_{i\in R}\bar{\pi}_i\geqslant 0$$

is nonnegative for each $R\in\mathcal{R}$. If the answer is yes, then $\bar{z}$ is an optimal solution to $(\mathcal{M}_{\mathcal{R}})$. If, on the other hand, a column (R,j) with negative reduced cost is found, then the column (R,j) is added to the master LP, and the next iteration begins.

Finding $R\subseteq\mathcal{R}$ and $j\in I$ with negative reduced cost, or proving that no such pair (R,j) exists, is called the *pricing problem.*

To solve the pricing problem, we need to solve for each fixed j, the following integer-programming problem:

$$(\mathcal{P}_j)\quad \min f_j+\sum_{i\in I}(\alpha_{ij}-\bar{\pi}_i)Y_{ij} + G_j\left(\sum_{i\in I}\mu_i Y_{ij}\right)+H_j\left(\sum_{i\in I}\sigma_i^2 Y_{ij}\right)$$

subject to

$$Y_{ij}\in\{0,1\} \quad \forall i\in I.$$

4. The Pricing Problem

In this section, we propose an algorithm to solve the pricing problem $(\mathcal{P}_j)$. To simplify the notation, we define

$$a_i:=\alpha_{ij}-\bar{\pi}_i,$$

$$b_i:=\mu_i,$$

$$c_i:=\sigma_i^2,$$

$$z_i:=Y_{ij}$$

for each $i\in I$. Note that f_j does not depend on Y_{ij} and, hence, can be ignored for discussion here. We now have the following problem $\mathcal{P}_j$ for designated distribution center $j\in I$:

$$(\mathcal{P}_j')\quad \min\sum_{i\in I}a_i z_i+G_j\left(\sum_{i\in I}b_i z_i\right)+H_j\left(\sum_{i\in I}c_i z_i\right)$$

subject to

$$z_i\in\{0,1\} \quad \forall i\in I.$$

For each $j\in I$, define set function g_j on I as follows: For each $S\subseteq I$,

$$g_j(S)\equiv\sum_{i\in S}a_i+G_j\left(\sum_{i\in S}b_i\right)+H_j\left(\sum_{i\in S}c_i\right). \tag{11}$$

The above problem can also be reformulated as

$$\min_{S\subseteq I} g_j(S).$$

Let z^* be an optimal solution to $(\mathcal{P}_j')$, with associated optimal objective value ω_j^*. The minimum reduced cost set $R_j^*\subset I$ is then the collection of retailers $i\in I$ with $z_i^*=1$. If $\omega_j^*+f_j\geqslant 0$, then the column (R_j^*,j) has nonnegative reduced cost. Hence, we can conclude that there is no set $R\in\mathcal{R}$ having designated distribution center j with negative reduced cost. Otherwise, we obtain a column with negative reduced cost.

Lemma 1. *Given a retailer $j\in I$, there exists a minimum reduced cost set $R_j^*\subset I$ with $a_i<0$ for every $i\in R_j^*$.*

Proof. Let $i\in R_j^*$. Because $b_i, c_i\geqslant 0$, and if $a_i\geqslant 0$, then for any solution $\bar{z}$ with $\bar{z}_i=1$, the objective function value is at least as great as that of the solution obtained from $\bar{z}$ by setting $\bar{z}_i=0$. This follows because G_j and H_j are assumed to be concave and nondecreasing. By repeating this process, we obtain a set R_j^* with the desired property. □

Hence, we may restrict our search for R_j^* to retailers in I^-, where $I^-:=\{i\in I: a_i<0\}$. We next identify a nice structural property of the set R_j^* by extending an argument in Chakravarty et al. (1985).

Let $a_S=\sum_{i\in S}a_i$, $b_S=\sum_{i\in S}b_i$, and $c_S=\sum_{i\in S}c_i$. Define a new function

$$h_j(x,y,z):=x+G_j(y)+H_j(z). \tag{12}$$

Note that $h_j(x,y,z)$ is a separable concave function, and

$$\min_{S\subseteq I^-}g_j(S)=\min_{S\subseteq I^-}h_j(a_S,b_S,c_S). \tag{13}$$

Because the set $\{(a_S,b_S,c_S): S\subseteq I^-\}$ is finite, its convex hull, which will be denoted by H, is a convex polyhedron. Now, because the function h_j is concave,

$$\min_{S\subseteq I^-}g_j(S)=\min_{S\subseteq I^-}h_j(a_S,b_S,c_S) =\min_{(a,b,c)\in H}h_j(a,b,c),$$

as the latter minimization problem attains a minimum at an extreme point of H.

Let (a^*,b^*,c^*) be an extreme point of H. Because H is a polyhedron, it is well known that there exists a linear function f on H that attains its *unique* minimum over H at (a^*,b^*,c^*). Because f is linear, it has a representation $f(a,b,c)=\alpha' a+\beta' b+\gamma' c$ defined by real numbers α', β', and γ'. The uniqueness of (a^*, b^*, c^*) as the minimizer of f over H assures that we do not have $\alpha' = \beta' = \gamma' = 0$.

Because H is the convex hull of $\{(a_S, b_S, c_S)\colon S \subseteq I^-\}$,

$$\begin{aligned}\alpha' a^* + \beta' b^* + \gamma' c^* &= \min_{(a,b,c)\in H} \alpha' a + \beta' b + \gamma' c\\ &= \min_{S\subseteq I^-} \alpha' a_S + \beta' b_S + \gamma' c_S\\ &= \min_{S\subseteq I^-} \sum_{i\in S} (\alpha' a_i + \beta' b_i + \gamma' c_i).\end{aligned}$$

The set $S^* = \{i \in I^-\colon \alpha' a_i + \beta' b_i + \gamma' c_i < 0\}$ is clearly optimal for the above optimization problem. Hence, we conclude from the uniqueness property that $(a^*, b^*, c^*) = (a_{S^*}, b_{S^*}, c_{S^*})$; i.e., $R_j^* = S^*$.

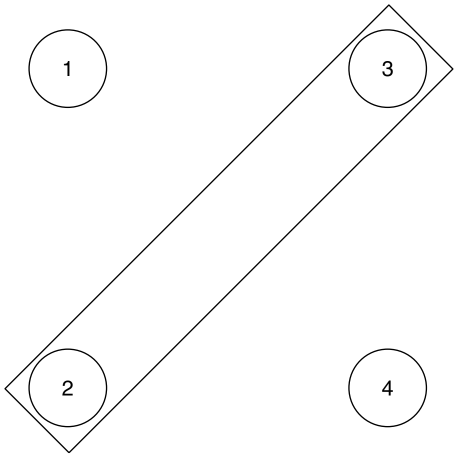
Note that $S^* = \{i\colon \alpha' a_i + \beta' b_i + \gamma' c_i < 0\} = \{i\colon \alpha' + \beta'(b_i/a_i) + \gamma'(c_i/a_i) > 0\} = \{i\colon \beta' x_i + \gamma' y_i < \alpha'\}$, where $x_i = -b_i/a_i$ and $y_i = -c_i/a_i$. Here, $x_i, y_i \geqslant 0$ for all i.

To determine S^*, we need to determine the corresponding values for α', β', and γ'. Although there are infinitely many choices for the parameters α', β', and γ', it turns out that the number of distinct partitions obtained by varying the parameters are limited. It is not difficult to see why the number of partitions obtained this way will not be large. Consider the following examples with four retailers (see Figure 1).

The position of the retailers in Figure 1 corresponds to the coordinate (x_i, y_i) given above. By varying the parameters α', β', and γ', we are looking for distinct partitions of the set of retailers with *half-space* in the plane. Note that instead of $2^4 = 16$ possible partitions, the number is restricted by the position of the points in the plane. For instance, Retailers 2 and 3 cannot be candidates, because it is not possible to separate them from Retailers 1 and 4 by using the intersection of a half-space and the set of points.

This phenomenon can be studied using a general result in the theory of VC-dimension (cf., Vapnik and Chervonenkis 1971). To describe this result, we need to first introduce some notations.

Figure 1. Partitioning.



The *VC-dimension* is defined for any set system $\mathcal{S} \subset 2^X$ on an arbitrary set X. It is the supremum of the sizes of all shattered subsets $\mathcal{A} \subset X$; here $\mathcal{A}$ is called shattered if $\mathcal{S}|_{\mathcal{A}} = 2^{\mathcal{A}}$; i.e., for any $\mathcal{B} \subset \mathcal{A}$ there exists a set $S \in \mathcal{S}$ such that $\mathcal{B} = \mathcal{A} \cap S$. For example, if $\mathcal{H}$ denotes the system of all closed half-planes in the plane, then it is not difficult to check that the VC-dimension of the set system $\mathcal{H}$ is three, because no four points in the plane can be shattered by using only half-planes.

The following well-known result shows that the number of possible candidates for S^* is essentially small:

Lemma 2 (Vapnik and Chervonenkis 1971, Sauer 1972). *For any set system $\mathcal{S}$ of VC-dimension at most d, we have $|\mathcal{S}|_X| \leqslant \Phi_d(|X|)$, where*

$$\Phi_d(m) = \binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{d}.$$

The above lemma suggests that we need to search among at most $O(n^3)$ possible subsets to determine S^*.

The complexity of the above algorithm can be slashed down further to $O(n^2)$, by enumerating over the extreme points of an associated zonotope (cf., Onn and Schulzman 2001) using an incremental algorithm. This algorithm, however, is not efficient in practice, because of the sophisticated data structure needed to implement the incremental algorithm. In the rest of this section, we show a more efficient and direct approach.

Algorithm for the Pricing Problem

Note that the parameters $(\alpha', \beta', \gamma')$ for the optimal S^* can be chosen to be the gradient function of $h_j(x, y, z) = x + G_j(y) + H_j(z)$ at the optimal solution (a^*, b^*, c^*). Hence, we can assume that $\alpha' = 1$, $\beta' \geqslant 0$, and $\gamma' \geqslant 0$. The problem reduces to finding S^* such that

$$S^* = \{i\colon \beta' x_i + \gamma' y_i < 1\}.$$

The case where $\beta' = 0$ or $\gamma' = 0$ is easy to handle and reduces to the special cases discussed in Shen et al. (2003). We assume next that $\beta' > 0$ and $\gamma' > 0$. Let

$$f_i = \beta' x_i + \gamma' y_i - 1.$$

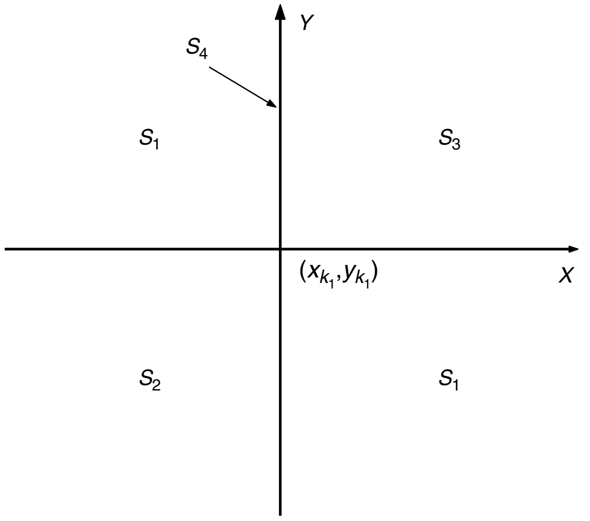
Let $k_1 = \arg\max\{f_i\colon f_i < 0\}$; i.e., k_1 corresponds to the retailer with the *largest negative* f_i. Note that S^* must now satisfy

$$\begin{aligned}S^* &= \{i\colon f_i \leqslant f_{k_1}\} \cup \{k_1\}\\ &= \{i\colon \beta'(x_i - x_{k_1}) + \gamma'(y_i - y_{k_1}) \leqslant 0\} \cup \{k_1\}\end{aligned}$$

for all $i \in I^- \backslash \{k_1\}$.

We partition the set $I^- \backslash \{k_1\}$ (say with cardinality L) into four different subsets, S_1, S_2, S_3, and S_4, with $|S_1| = m_1$, $|S_2| = m_2$, $|S_3| = m_3$, and $|S_4| = L - m_1 - m_2 - m_3$. For retailer $i \in I^- \backslash \{k_1\}$, if $(x_i - x_{k_1})/(y_i - y_{k_1}) \leqslant 0$, then i

Figure 2. Different subsets.



belongs to set S_1. If $(x_i - x_{k_1})/(y_i - y_{k_1}) > 0$ and $x_i < x_{k_1}$, $y_i < y_{k_1}$, then retailer i belongs to set S_2; if $(x_i - x_{k_1})/(y_i - y_{k_1}) > 0$ and $x_i > x_{k_1}$, $y_i > y_{k_1}$, then retailer i belongs to set S_3. All the remaining retailers in $I^- \backslash \{k_1\}$ belong to set S_4. It is easy to see that $y_i = y_{k_1}$ for $i \in S_4$. See Figure 2 for a graphical illustration of the four subsets.

We further sort the retailers in subset S_1 by relabelling the indices if necessary, so that

$$\frac{x_1 - x_{k_1}}{y_1 - y_{k_1}} \leqslant \frac{x_2 - x_{k_1}}{y_2 - y_{k_1}} \leqslant \cdots \leqslant \frac{x_{m_1} - x_{k_1}}{y_{m_1} - y_{k_1}} \leqslant 0. \qquad (*)$$

Because $\beta' > 0$ and $\gamma' > 0$, $i \in S^*$ for all i in S_2 and $i \notin S^*$ for all i in S_3. For $i \in S_4$, because $\beta' > 0$, $i \in S^*$ if and only if $x_i - x_{k_1} \leqslant 0$.

To determine whether $i \in S^*$ if $i \in S_1$, first note that

$$\frac{x_i - x_{k_1}}{y_i - y_{k_1}} \leqslant \frac{x_j - x_{k_1}}{y_j - y_{k_1}} \quad \text{if and only if}$$

$$\beta' \frac{x_i - x_{k_1}}{y_i - y_{k_1}} + \gamma' \leqslant \beta' \frac{x_j - x_{k_1}}{y_j - y_{k_1}} + \gamma'.$$

We consider the following three cases:

Case (a). Suppose that there exists a retailer k_2 such that the (open) interval

$$\left(\beta' \frac{x_{k_2} - x_{k_1}}{y_{k_2} - y_{k_1}} + \gamma', \beta' \frac{x_{k_2+1} - x_{k_1}}{y_{k_2+1} - y_{k_1}} + \gamma'\right)$$

contains the point 0.

Claim 1. *If $i \leqslant k_2$, then $i \in S^*$ if and only if $y_i - y_{k_1} > 0$.*

Proof. Because

$$\beta' \frac{x_i - x_{k_1}}{y_i - y_{k_1}} + \gamma' \leqslant \beta' \frac{x_{k_2} - x_{k_1}}{y_{k_2} - y_{k_1}} + \gamma' < 0,$$

we have

$$\beta'(x_i - x_{k_1}) + \gamma'(y_i - y_{k_1}) < 0$$

if and only if $y_i - y_{k_1} > 0$. □

Similarly,

Claim 2. *If $m_1 \geqslant i \geqslant k_2 + 1$, then $i \in S^*$ if and only if $y_i - y_{k_1} < 0$.*

Proof. Because

$$\beta' \frac{x_i - x_{k_1}}{y_i - y_{k_1}} + \gamma' \geqslant \beta' \frac{x_{k_2+1} - x_{k_1}}{y_{k_2+1} - y_{k_1}} + \gamma' > 0,$$

we have

$$\beta'(x_i - x_{k_1}) + \gamma'(y_i - y_{k_1}) < 0$$

if and only if $y_i - y_{k_1} < 0$. □

Case (b). Suppose that there exists a retailer k_2 such that $\beta'(x_{k_2} - x_{k_1})/(y_{k_2} - y_{k_1}) + \gamma' = 0$. Then, clearly, k_2 satisfies $f_{k_2} \leqslant f_{k_1}$ and hence $k_2 \in S^*$. Furthermore, all i with

$$\frac{x_i - x_{k_1}}{y_i - y_{k_1}} = \frac{x_{k_2} - x_{k_1}}{y_{k_2} - y_{k_1}}$$

are also candidates for S^* because they all satisfy $f_i = f_{k_2} \leqslant f_{k_1}$.

Case (c). Suppose there is no retailer k_2 that satisfies Case (a) or Case (b), i.e., either

$$0 < \beta' \frac{x_1 - x_{k_1}}{y_1 - y_{k_1}} + \gamma' \quad \text{or} \quad 0 > \beta' \frac{x_{m_1} - x_{k_1}}{y_{m_1} - y_{k_1}} + \gamma'.$$

Then, conditions similar to the two situations discussed in Case (a) can be used to determine whether $i \in S^*$. □

Therefore, we obtain the following characterization for the set S^*:

Theorem 1. *The optimal solution S^* satisfies the following properties for some k_1 and k_2, and with the retailers ordered as in $(*)$:*

1. *$k_1 \in S^*$.*
2. *For all $i \in S_1$, either Case* (a) *holds*:
   - *for all $i \in \{1, 2, \ldots, k_2\}$, $i \in S^*$, if and only if $y_i - y_{k_1} > 0$;*
   - *for all $i \in \{k_2 + 1, \ldots, m_1\}$, $i \in S^*$, if and only if $y_i - y_{k_1} < 0$;*

   or Case (b) *holds*:
   - *for all $i < k_2$, with*

$$\frac{x_i - x_{k_1}}{y_i - y_{k_1}} < \frac{x_{k_2} - x_{k_1}}{y_{k_2} - y_{k_1}}, \quad i \in S^*,$$

if and only if $y_i - y_{k_1} > 0$;
   - *for all $i > k_2$, with*

$$\frac{x_i - x_{k_1}}{y_i - y_{k_1}} > \frac{x_{k_2} - x_{k_1}}{y_{k_2} - y_{k_1}}, \quad i \in S^*,$$

if and only if $y_i - y_{k_1} < 0$;
   - *for i such that*

$$\frac{x_i - x_{k_1}}{y_i - y_{k_1}} = \frac{x_{k_2} - x_{k_1}}{y_{k_2} - y_{k_1}}, \quad i \in S^*.$$

3. *For all* $i \in S_2$, $i \in S^*$;
4. *For all* $i \in S_3$, $i \notin S^*$;
5. *For all* $i \in S_4$, $i \in S^*$, *if and only if* $x_i - x_{k_1} \leqslant 0$.

The above theorem provides an efficient method for finding S^*, thus solving the pricing problem $\mathcal{P}_j$. Although k_1 and k_2 are not given a priori, we can simply guess k_1 from 1 to n and k_2 from 1 to m_1 (after sorting to satisfy $(*)$). For each pair of k_1 and k_2, we can easily generate *all the solutions* satisfying the above properties and then select the one with the lowest objective value. It is easy to see that for each specific k_1, there are at most $2n$ such solutions. They can be listed immediately after sorting the values $(x_i - x_{k_1})/(y_i - y_{k_1})$. With a bit of reflection, it is easy to see that we are able to compute the entire list of the candidate solution for S^* with $O(n)$ multiplications, additions, and square-root computations. Thus, sorting, which can be done in time $O(n \log n)$, is the dominant step in this algorithm. We need to do this sorting n times, one for each k_1 ranging from 1 to n, so the computational complexity of this algorithm is $O(n^2 \log n)$.

Theorem 2. *The problem* $\min_{S \subseteq I^-} g_j(S)$ *can be solved in* $O(n^2 \log n)$ *time.*

5. Variable Fixing

In the straightforward implementation of the above algorithm, we need to solve, for each retailer, a related submodular function minimization problem where the retailer is assumed to be the DC. This slows down the column generation routine considerably. We show next how information on the primal and dual solution can be used to "fix" variables, so that we can determine whether a retailer will be a DC candidate in an optimal solution early in the column generation routine.

Recall that the set-covering model we are trying to solve is of the form

$$\min \sum_{R \in \mathcal{R}} \sum_{j \in I} c_{R,j} z_{R,j}$$

subject to

$$\sum_{R \in \mathcal{R}: i \in R} \left(\sum_{j \in I} z_{R,j} \right) \geqslant 1 \quad \forall i \in I,$$

$$z_{R,j} \in \{0, 1\} \quad \forall R \in \mathcal{R}.$$

At each stage of the column generation routine, we have

- A set of dual prices $\{\pi_j\}$.
- A set of primal feasible (fractional) solution $z_{S,j}$.
- After solving the pricing problem (one for each potential DC location), we obtain the reduced cost $r_j \equiv \min_{S: j \in S}(c_{S,j} - \sum_{k \in S} \pi_k)$. Note that some of the r_js may be nonnegative.

Let Z_{IP} and Z_{LP} denote the optimal integral and fractional solution to the set-covering problem.

Claim 3. $\sum_{j: r_j \leqslant 0} r_j + \sum_j \pi_j$ *is a lower bound to* Z_{IP}.

Proof. In the optimal IP solution, by concavity of the objective function, it is easy to check that $c_{S,j} + c_{T,j} \leqslant c_{S \cup T, j}$. Hence, the inequality

$$\sum_S z_{S,j} \geqslant 1$$

is a valid inequality for the problem. We can add the above inequality, one for each j, to the set-covering model, to obtain a stronger LP relaxation. The Lagrangian dual of the new LP relaxation is thus equivalent to

$$L(\lambda) = \sum_j \lambda_j + \min \left\{ \left(\sum_j \sum_S \left(c_{S,j} - \sum_{k \in S} \lambda_k \right) z_{S,j} \right) : \right.$$
$$\left. 1 \geqslant z_{S,j} \geqslant 0, \sum_S z_{S,j} \leqslant 1 \ \forall j \right\}.$$

The problem decomposes for each retailer j, and hence $Z_{\mathrm{IP}} \geqslant \max_\lambda L(\lambda) \geqslant L(\pi) = \sum_j \pi_j + \sum_{j: r_j \leqslant 0} r_j$. □

Let j^* be a retailer such that $r^*_j > 0$. Let UB be an upper bound for Z_{IP}.

Claim 4. *If* $\sum_{j: r_j \leqslant 0} r_j + \sum_j \pi_j + r_{j^*} > UB$, *then retailer* j^* *will never be used as a DC in the optimal solution to the (integral) set-covering problem.*

Proof. To see this, suppose otherwise. Then, Z_{IP} remains unchanged if we impose the additional condition $\sum_S z_{S,j^*} = 1$ to the existing set of constraints. The Lagrangian dual, in this case, reduces to

$$L'(\lambda)$$
$$= \sum_j \lambda_j + \min \left\{ \left(\sum_j \sum_{S: j \in S} \left(c_{S,j} - \sum_{k \in S} \lambda_k \right) z_{S,j} \right) : 1 \geqslant z_{S,j} \geqslant 0, \right.$$
$$\left. \sum_S z_{S,j} \leqslant 1 \ \forall j, \sum_S z_{S,j^*} = 1 \right\}.$$

Hence, $Z_{\mathrm{IP}} \geqslant \max_\lambda L'(\lambda) \geqslant L'(\pi) = \sum_j \pi_j + \sum_{j: r_j \leqslant 0} r_j + r_{j^*}$. On the other hand, we have $Z_{\mathrm{IP}} \leqslant \mathrm{UB}$. This gives rise to a contradiction. □

Note that once we determine that the retailer j^* will never be used as a DC in the optimal solution, then we do not need to solve the pricing problem corresponding to j^* anymore in the rest of the column generation procedure. In fact, all columns arising from using j^* as a DC (generated previously) can also be deleted from the LP. This is the key advantage of the variable fixing method.

The variable fixing method depends largely on the quality of the upper bound UB. If $Z_{\mathrm{LP}} = Z_{\mathrm{IP}}$, then the solution $\sum_{S,j} c_{S,j} z_{S,j}$ generated at each stage of the column generation routine will be an upper bound to Z_{IP}. Unfortunately, this is not true for all instances. As in Daskin et al. (2002), we generate an upper bound for the IP by generating a feasible solution in the following way:

- Let z^* be the optimal LP solution obtained by solving the problem using a partial set of columns.

- Order the retailers according to nondecreasing value of demand.
- Starting from the first retailer (say i) on the list, if for some S and j, $i \in S$ and $z^*_{S,j} = 1$, then retailer i is served by DC j. Otherwise, there exist S, T, both containing i, and j, k, such that $z^*_{S,j} > 0$ and $z^*_{T,k} > 0$. We serve i using the DC that will lead to the least total cost, and remove retailer i from the list.
- Repeat the previous step until the list is empty.

In this way, we can generate a feasible solution to the distribution network design problem. This solution will be used as a bound to perform variable fixing in the column generation routine.

6. Computational Results

In this section, we summarize our computational experience with the algorithms outlined in the previous section. All the instances were solved on a COMPAQ P3-450 station running the Windows 2000 operating system. We used the risk-pooling model as described in Shen et al. (2003) to design the computational experiment for our model. To facilitate proper comparison, we have also imposed the additional assumption, used in Shen et al. (2003), that a DC at retailer j must be used to serve the demand at retailer j. Note that this is not necessarily true in the optimal solution. The solution approach described for the general case can be easily modified to handle this additional assumption. We leave the details to the readers.

6.1. Submodular Function Minimization

The purpose of this section is to test the performance-of-pricing algorithm. The code is written in C++. α is randomly generated in $[50\%, 100\%]$, β is generated uniformly in $[0.001, 0.01]$, θ is randomly generated in $(0, 20]$, and all the other parameters are generated uniformly in $[0, 100]$.

Table 1 presents the relation between the average CPU time needed (averaged over 20 different instances) and the number of retailers in the problem. For each instance, we vary the DC choices over all the possible retailer locations. The CPU time we report is the total running time for solving all the pricing problems with each retailer as the DC at a time.

Up to 320 retailers, each pricing problem can be solved in less than two seconds. This shows that the pricing problem for moderate to large-size problems can be solved efficiently using this method.

Table 1. CPU time of the pricing algorithm.

No. of retailers	CPU time (seconds)
10	0.01
20	0.07
40	0.73
80	6.91
160	60.3
320	554

6.2. Stochastic Network Design Without Variable Fixing

In this subsection, we report the results of solving the network design problem using the column generation method. The algorithm for the general network distribution problem is coded in C++, and the LP problem is solved using CPLEX LP Solver.

The mean demands μ_i and σ_i^2 are randomly generated in $[100, 1600]$ for all $i \in I$. Holding cost is 1, $z_\alpha = 1.96$ (97.5% service level), $a_i = 5$, $g_i = 10$, and $F_i = 10$ for all $i \in I$. Our goal is to find ranges of values for β and θ that resulted in instances that varied in solution difficulty as well as the fraction of retailers used as DCs in the solution.

For each of the instances, we first solve the LP relaxation of the set-covering model via column generation. The initial set of columns includes all singletons. The column labelled "No. of columns generated" indicates the total number of columns added during this phase. The resulting final optimal objective value is denoted by Z_{LP}. In most instances generated, the corresponding optimal solutions are integral. We denote by Z_H the best upper bound we obtained. In the case where the LP relaxation solution is not integral, Z_H is obtained by applying an IP solver to the final master problem.

To speed up the column generation algorithm, we do not solve the pricing problem at every iteration. Instead, we maintain a column pool and we first look for columns with negative reduced cost in the column pool. If the search is successful, we add these columns to the master problem. We also update the reduced costs of the columns in the column pool and remove those with large positive reduced costs from the pool. We solve the pricing problem only when the column pool is empty. When this happens, the pricing algorithm is run to either prove the optimality of the current solution or to find new columns with negative reduced costs.

Tables 2, 3, and 4 show the results of our computational study.

The above algorithm becomes very time consuming when the number of retailers exceeds 120. For this reason, we do not report computational results for larger instances. To tackle a larger-scale problem, we turn to the variable fixing method.

6.3. Stochastic Network Design with Variable Fixing

The column labelled "No. of DCs Out" indicates the number of retailers ruled out from being possible DCs in the optimal LP solution by the variable fixing technique. The parameters we use for the instances below are the same as what we used for the previous subsection. Only when the column pool is empty shall we do the variable fixing procedure again. Tables 5, 6, 7, 8, and 9 highlight the results of our computational study.

Table 2. Computational results for the 40-retailer instance.

Input		Output			
β	θ	No. of DCs opened	CPU time	No. of columns generated	Z_H/Z_{LP}
0.001	0.1	5	62	2,661	1
0.002	0.1	7	16	961	1.001
0.003	0.1	8	7	543	1
0.004	0.1	10	6	466	1
0.005	0.1	14	5	364	1
0.001	0.1	5	62	2,661	1
0.002	0.2	6	16	974	1.001
0.005	0.5	9	6	492	1
0.005	0.1	14	5	364	1
0.005	1	11	6	430	1.003
0.005	5	8	9	828	1
0.005	10	7	28	1,643	1.001

Table 3. Computational results for the 80-retailer instance.

Input		Output			
β	θ	No. of DCs opened	CPU time	No. of columns generated	Z_H/Z_{LP}
0.001	0.1	7	414	9,984	1
0.002	0.1	9	201	6,031	1
0.003	0.1	12	94	2,548	1
0.004	0.1	21	47	1,042	1
0.005	0.1	24	26	511	1
0.001	0.1	7	414	9,984	1
0.002	0.2	13	84	2,077	1
0.005	0.5	18	52	1,196	1
0.005	0.1	24	26	511	1
0.005	1	10	102	3,096	1
0.005	10	8	364	9,072	1

Table 4. Computational results for the 120-retailer instance.

Input		Output			
β	θ	No. of DCs opened	CPU time	No. of columns generated	Z_H/Z_{LP}
0.0001	0.01	11	742	21,537	1
0.0002	0.01	15	516	11,476	1
0.0003	0.01	24	213	4,997	1.001
0.0004	0.01	28	103	2,014	1
0.0005	0.01	33	57	1,043	1
0.0001	0.01	11	742	21,537	1
0.0002	0.02	23	215	5,021	1
0.0005	0.05	29	99	1,998	1
0.0005	0.01	33	57	1,043	1
0.0005	0.1	15	522	11,628	1
0.0005	0.5	12	635	17,142	1.002

Table 5. Computational results for the 40-retailer instance with variable fixing.

Input		Output				
β	θ	No. of DCs opened	No. of DCs out	CPU time	No. of columns generated	Z_H/Z_{LP}
0.001	0.1	5	34	6	117	1
0.002	0.1	7	31	3	73	1
0.003	0.1	8	31	2	66	1
0.004	0.1	10	29	1	44	1.001
0.005	0.1	13	27	1	32	1
0.001	0.1	5	34	6	117	1
0.002	0.2	6	34	4	92	1.001
0.005	0.5	8	32	2	56	1
0.005	0.1	13	27	1	32	1
0.005	1	10	29	1	40	1.002
0.005	5	7	32	4	96	1
0.005	10	5	34	6	122	1.001

Table 6. Computational results for the 80-retailer instance with variable fixing.

Input		Output				
β	θ	No. of DCs opened	No. of DCs out	CPU time	No. of columns generated	Z_H/Z_{LP}
0.001	0.1	6	74	31	348	1
0.002	0.1	8	72	20	202	1
0.003	0.1	12	67	14	147	1
0.004	0.1	21	58	9	84	1
0.005	0.1	24	56	7	66	1
0.001	0.1	6	74	31	348	1
0.002	0.2	13	66	14	142	1
0.005	0.5	18	62	10	113	1
0.005	0.1	24	56	7	66	1
0.005	1	10	69	15	162	1
0.005	10	7	72	27	303	1

Table 7. Computational results for the 120-retailer instance with variable fixing.

Input		Output				
β	θ	No. of DCs opened	No. of DCs out	CPU time	No. of columns generated	Z_H/Z_{LP}
0.0001	0.01	10	109	94	804	1
0.0002	0.01	15	105	61	480	1
0.0003	0.01	24	94	38	296	1
0.0004	0.01	28	90	27	184	1
0.0005	0.01	33	87	16	97	1
0.0001	0.01	10	109	94	804	1
0.0002	0.02	23	96	38	309	1
0.0005	0.05	29	90	26	166	1
0.0005	0.01	33	87	16	97	1
0.0005	0.1	15	104	62	496	1
0.0005	0.5	11	109	90	743	1.002

Table 8. Computational results for the 250-retailer instance with variable fixing.

Input		Output				
β	θ	No. of DCs opened	No. of DCs out	CPU time	No. of columns generated	Z_H/Z_{LP}
0.0001	0.01	19	229	221	1,663	1
0.0002	0.01	30	219	113	782	1
0.0003	0.01	49	201	78	466	1.002
0.0004	0.01	57	192	59	346	1
0.0005	0.01	68	182	35	201	1
0.0001	0.01	19	229	221	1,663	1
0.0002	0.02	44	206	83	526	1.001
0.0005	0.05	61	189	47	302	1
0.0005	0.01	68	182	35	201	1
0.0005	0.1	33	215	118	697	1
0.0005	0.5	20	230	211	1,479	1

By applying the variable fixing technique, we are able to cut down the computational time dramatically. The average CPU time is only about 9% of the CPU time without the variable fixing technique. The savings range from about 83% to 94%. It is especially effective for those difficult instances that required the most CPU times before. For example, for $\beta = 0.001$, $\theta = 0.1$ in the 80-retailer case, we are able to solve the problem in about 30 seconds after applying the variable fixing technique, which used to take 414 seconds.

As we can see from these tables, the problem takes longer to solve when β decreases for fixed θ or when θ increases for fixed β. When the transportation costs increase relative to other costs, i.e., β increases, the "No. of DCs opened" increases too. However, when the inventory costs increase relative to other costs, i.e., θ increases, the "No. of DCs opened" decreases.

Table 9. Computational results for the 500-retailer instance with variable fixing.

Input		Output				
β	θ	No. of DCs opened	No. of DCs out	CPU time	No. of columns generated	Z_H/Z_{LP}
0.0001	0.01	42	458	512	3,742	1
0.0002	0.01	57	442	426	2,819	1.001
0.0003	0.01	95	404	248	1,405	1
0.0004	0.01	114	386	146	717	1
0.0005	0.01	146	354	86	446	1
0.0001	0.01	42	458	512	3,742	1
0.0002	0.02	90	409	314	1,833	1
0.0005	0.05	132	368	117	586	1
0.0005	0.01	146	354	86	446	1
0.0005	0.1	61	439	404	2,633	1
0.0005	0.5	44	455	503	3,572	1

7. Extensions

7.1. Distance Constraint

In the distribution network design problem, it is quite common to impose additional constraints on the collection of retailers a DC can serve. For example, a typical geographical constraint stipulates that the designated DC and the retailer cannot be too far apart. To enforce this constraint in our approach is easy, because we can set the distance function d_{ij} to a huge number if retailer j cannot act as the DC for retailer i (or vice versa).

7.2. Capacity Constraint

Another common constraint states that a DC cannot handle too many retailers (say not more than k retailers can be served by a single DC), due to capacity or other technical limitations. In this paper, we describe how our technique can be extended to handle the additional constraint of the type $\sum_i Y_{i,j} \leqslant k$ for some fixed k. Note that we are assuming that retailer j needs not be served by the DC located at its location.

In this case, the column generation phase reduces to solving a problem of the type

$$\min \sum_{i\in I} a_i z_i + G_j\left(\sum_{i\in I} b_i z_i\right) + H_j\left(\sum_{i\in I} c_i z_i\right)$$

subject to

$$z_i \in \{0,1\} \quad \forall i \in I,$$

$$z_j = 1,$$

$$\sum_i z_i \leqslant k.$$

Because the objective function is concave and separable, we can use the same argument to reduce the problem to a parametric version:

$$\min_{S\subseteq I^-} \sum_{i\in S} (\alpha' a_i + \beta' b_i + \gamma' c_i) z_i$$

$$(PK)\text{: subject to}$$

$$z_i \in \{0,1\} \quad \forall i \in I^-,$$

$$\sum_{i\in I^-} z_i \leqslant k.$$

The candidate solution for the column generation phase comes out as the solution to the above linear discrete optimization problem for some choice of α', β', and γ'.

Let $b(\alpha', \beta', \gamma')$ denote the value of the kth smallest entry in the set

$$\{\alpha' a_i + \beta' b_i + \gamma' c_i \colon i \in I^-\}.$$

It is clear that if $z_i = 1$ in an optimal solution to problem (PK), then clearly $\alpha' a_i + \beta' b_i + \gamma' c_i \leqslant b(\alpha', \beta', \gamma')$ because $\alpha' a_i + \beta' b_i + \gamma' c_i$ cannot be bigger than the kth smallest value. Furthermore, we need $\alpha' a_i + \beta' b_i + \gamma' c_i < 0$, otherwise we would have $z_i = 0$ in the optimal solution. Conversely, it is easy to see that $z_i = 1$ in the optimal solution if the point i satisfies both inequalities.

The inequality

$$\alpha' a_i + \beta' b_i + \gamma' c_i < \min(b(\alpha', \beta', \gamma'), 0)$$

determines a half-plane in 3D, and at most k out of possible $n-1$ points in the set

$$S \equiv \{(a_i, b_i, c_i) \colon i \in I^-\}$$

lies in this half-plane.

Hence, the number of candidate solutions depends on the number of $\leqslant k$-set. Here, a $\leqslant k$-set is the intersection of S and a half-plane containing at most k points. Clarkson and Shor (1989) showed that the number of such solutions in 3D is bounded above by $O(nk^2)$. Hence, the number of candidate sets is still bounded by a polynomial in n.

8. Conclusion

In this paper, we have outlined a formulation of a stochastic transportation-inventory network design model. The model determines how many and where to locate regional DCs and how to assign retailers to the DCs to minimize the total system costs, which include DC location costs, inventory costs at the DCs, and the transportation costs within this two-echelon supply chain.

The model was originally proposed in Shen et al. (2003). They were able to solve efficiently only two special cases of the general model. We proposed an efficient algorithm to solve the general pricing problem, with a worst-case running time of $O(n^2 \log n)$. Together with the variable fixing technique, this yields a very efficient approach to solve a moderate to large-scale network design problem to near optimality.

We would like to emphasize the importance of being able to solve the general supply chain design problem. The two cases considered in Shen et al. (2003) require that the demand be either deterministic or $\sigma_i^2/\mu_i = \gamma$ for every retailer. However, in a lot of real-life situations, the demand processes can be very different from retailer to retailer, and the ratio of demand variance to mean demand are not the same for different retailers. Supply chain network design problems under such conditions are the ones that the management is most concerned with, and our model can be applied successfully in the decision-making process for problems of this kind. Furthermore, we show that our solution techniques can handle a more general risk-pooling type of network design problem, because our algorithm uses only the concavity property of the objective function.

We propose two important related future research directions. First, we believe that the network design problem when each DC operates under the optimal (Q, r) policy is worth exploring. This model captures the stochastic inventory replenishment cost at the DC using a more accurate cost model. The challenge in this problem, we believe, lies in showing that the optimal cost of a (Q, r) policy is concave in the average demand assigned. Second, we hope to consider the cases with multiple items as well as more general capacity constraints, and with more realistic transportation cost structures.

Acknowledgments

The authors thank the associate editor and two referees for constructive comments that led to this improved version of this paper. This work was supported in part by NSF grant DMI-0223323. This support is gratefully acknowledged. The work of Zuo-Jun Max Shen was done while he was with the University of Florida.

References

Barahona, F., D. Jensen. 1998. Plant location with minimal inventory. *Math. Programming* **83** 101–111.

Chakravarty, A. K., J. B. Orlin, U. G. Rothblum. 1985. Consecutive optimizers for a partitioning problem with applications to optimal inventory groupings for joint replenishment. *Oper. Res.* **33** 820–834.

Chopra, S., P. Meindl. 2001. *Supply Chain Management: Strategy, Planning and Operation.* Pearson Prentice Hall, Upper Saddle River, NJ.

Clarkson, K. L., P. W. Shor. 1989. Applications of random sampling in computational geometry. II. *Discrete Comput. Geometry* **4**(5) 387–421.

Current, J., M. S. Daskin, D. Schilling. 2002. Discrete network location models. Z. Drezner, H. Hamacher, eds. *Facility Location Theory: Applications and Methods*, Chap. 3. Springer-Verlag, Berlin, Germany, 81–118.

Daskin, M. S. 1995. *Network and Discrete Location: Models, Algorithms, and Applications.* Wiley-Interscience, New York.

Daskin, M. S., Collette R. Coullard, Z. J. Max Shen. 2002. An inventory-location model: Formulation, solution algorithm and computational results. *Annals Oper. Res.* **110** 83–106.

Drezner, Z., ed. 1995. *Facility Location: A Survey of Applications and Methods.* Springer, New York.

Eppen, G. 1979. Effects on centralization on expected costs in a multi-location newsboy problem. *Management Sci.* **25**(5) 498–501.

Erlebacher, S. J., R. D. Meller. 2000. The interaction of location and inventory in designing distribution systems. *IIE Trans.* **32** 155–166.

Geoffrion, A. M., R. Power. 1995. Twenty years of strategic distribution system design: An evolutionary perspective. *Interfaces* **25**(5) 105–127.

Geunes, J., Z. J. Shen, H. E. Romeijn. 2004. Economic ordering decisions with market choice flexibility. *Naval Res. Logist.* **51**(1) 117–136.

Graves, S. C., A. H. G. Rinnooy Kan, P. H. Zipkin. 1993. *Logistics of Production and Inventory.* Elsevier Science Publishers, Amsterdam, The Netherlands.

Grötschel, M., L. Lovasz, A. Schrijver. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* **1** 169–197.

Iwata, S., L. Fleischer, S. Fujishige. 2001. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. *J. ACM* **48** 761–777.

Mirchandani, P. B., R. L. Francis. 1990. *Discrete Location Theory.* John Wiley and Sons, New York.

Nahmias, S. 1997. *Production and Operations Management*, 3rd ed. Irwin, Chicago, IL.

Onn, S., L. Schulman. 2001. The vector partition problems for convex objective functions. *Math. Oper. Res.* **26** 583–590.

Rado, R. 1942. A theorem on independence relations. *Quart. J. Math. Oxford* **213** 83–89.

Sauer, N. 1972. On the density of families of sets. *J. Combinatorial Theory Series A* **1** 145–1472.

Schrijver, A. 2000. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *J. Combinatorial Theory, Series B* **80** 346–355.

Shen, Z. J. Max. 2000. Approximation algorithms for various supply chain problems. Ph.D. thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.

Shen, Z. J. Max, C. Coullard, M. S. Daskin. 2003. A joint location-inventory model. *Transportation Sci.* **37** 40–55.

Teo, C. P., J. H. Ou, K. H. Goh. 2001. Impact on inventory costs with consolidation of distribution centers. *IIE Trans.* **33**(2) 99–110.

Vapnik, V. N., A. Ya. Chervonenkis. 1971. On the uniform convergence of relative frequencies of events to their probabilities. *Theory Probab. Appl.* **16** 264–280.

Zipkin, P. H. 1997. *Foundations of Inventory Management.* Irwin, Burr Ridge, IL.